PD030127 US

(19) [Logo] European Patent Office (11) Publication No.: 0 379 436 A1

(12) EUROPEAN PATENT APPLICATION

(21) Serial number: 90400148.4 (51) Int. Cl.[5]: G06F 13/18

(22) Filing date: January 19, 1990

(30) Priority: January 20, 1989 FR 8900689

(43) Publication date of application: July 25, 1990, Bulletin 90/30

(84) Designated contracting states: DE GB

(71) Applicant: CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE (CNRS)
13 Quai Anatole France
F-75700 Paris (FR)

Applicant: UNIVERSITE PIERRE ET MARIE CURIE PARIS VI
4, place Jussieu
F-75252 Paris Cédex 05 (FR)

(72) Inventor: De Dinechin, Benoît
84, Boulevard Raspail
F-75006 Paris (FR)

(74) Agent: Rodhain, Claude, et al.
Cabinet Claude Rodhain
30, rue la Boétie
F-75008 Paris (FR)

(54) System and method for arbitrating requests and resolving conflicts related to access to independent-bank memories for computing machines

(57) The invention concerns an arbitration system and method intended for machines comprising at least one processor and a memory equipped with logical banks (BLO) linked by independent, simultaneously operating channels ($C_0, C_1 \ldots C_{p-1}$).

According to the invention, on the path of each logical bank and for each channel, a check is performed by a decoder (DEO_0) to determine whether the incoming request is intended for that logical bank, and if so, the attributes of the request are stored in a buffer (TLO_0) managed by a read/write manager (GEO_0); then, in an arbiter (ARO) common to all the channels providing access to that logical bank, a request is selected that can be passed to memory without triggering a conflict and is sent to that memory; and in the absence of an unsatisfied request having an absolute priority the priorities of the pending requests are updated, whereas in the presence of such a request a freeze is placed on the processor concerned.

System and method for arbitrating requests and resolving conflicts related to access to independent-bank memories for computing machines

The invention concerns a system and a method for arbitrating requests and resolving conflicts related to access to independent-bank memories for computing machines, and more particularly for high-performance parallel machines such as mini-supercomputers and image synthesizers.

Such machines comprise one or more processors connected to a shared central memory, and the entire machine is synchronized by a single clock. Each processor is connected to the memory by a plurality of access channels, which are independent and are able to operate simultaneously to each transfer up to one data item per clock cycle.

This is true of, notably, industrial supercomputers, signal processors, and most image synthesizers and computing machines equipped with more than one processor (the term "processor" here denoting any automaton capable of autonomously managing a stream of requests over one or more memory channels, i.e., with equal applicability, computing units of the machine, input/output controllers, or cache managers). This is because the volume of data processed by a supercomputer or in an image synthesizer demands the use of a large-capacity central memory (of more than a gigabyte in some cases); considerations of machine cost and reliability lead to the use of a denser and therefore slower technology for the memory circuits than for the processors, and the resulting speed discrepancy is exacerbated by the unilateral use (on the processor side) of pipeline techniques. Thus, the only way to get close to the ideal rate of one word per cycle and per channel is to use a parallel memory structure. In this configuration, the data are distributed among N independent memories known as physical banks. Taking the clock period as the time unit, the theoretical passband, in superwords per cycle, of a multi-bank memory in which each bank can service no more than one request per Tc cycles is therefore equal to INF(N/Tc,p), with p denoting the total number of channels.

There is a large difference between the theoretical passband of a multi-bank memory and its actual passband. Because of this difference, it is currently necessary to overdimension the number of physical banks, the factor (N/Tc)/p being on the order of 4 to 6 in many present-day machines, thus limiting by the same factor the maximum number of memory channels and consequently the machine parallelism that can be achieved.

One object of the invention is to remedy this drawback and thereby reduce the aforesaid difference that exists in machines with multi-bank memories through appropriate management of access to the memory system.

Before other objects to be achieved and problems to be solved by the invention are set forth, it may be noted by way of review that in known multi-bank memory systems the smallest data unit that can be accessed individually from a processor, which is known as a word, is usually in the form of a byte; however, 32- or 64- bit words are also commonly encountered; the maximum number of bits that a channel can transfer during each cycle, known as a superword, is 64, 128 or even as many as 512; the smallest physical unit of word-organized storage is known as a module; and a physical bank is composed of the number of modules necessary to form a superword from words; in some machines a superword is the same size as a word, and a physical bank is therefore identical to a module.

Assuming that the total number of channels p, the number of physical banks N, and the duration Tc of the cycles of the physical banks are given, a great deal of latitude still exists in organizing the memory system of a machine even if the design is limited to a specific type of application; but an important choice factor lies in the connection between the physical banks and the channels leading from the processors, where a compromise must be reached between the complexity, the latency and the conflict rate of the network; the best solution at present, essentially for reasons of latency reduction and simplicity of control, seems to be the use of systems known as full or partial "crossbar" systems. For instance, supercomputers are known in which the memory is organized into four sections of eight physical banks each, each of them offering one

access line per processor; a 3 x 4 crossbar network connects the three channels of each processor to the four associated access lines providing access to the sections.

A second choice factor comes into play in the resolution of network or bank conflicts, since the presence of modules that are unable to service more than one request per Tc cycles, simultaneously operating channels, and/or a connection network that has limited capacity raises the problem of memory system management. Such a conflict causes all the conflicting requests but one to be placed on hold, and the difficulty lies in choosing a decision algorithm that is effective in terms of outcome and is fast enough to keep pace with cycle time.

A third factor that has to be considered, since it has a significant impact on the performance of a memory system, is address distribution among the banks, an element that generally cannot be optimized for all applications of the machine. For example, according to some analyses, in the case of scientific calculations the access sequences on consecutive physical banks account for about 70% of memory references, leading to the generalized use of low-address interleaving. In this scheme, the address data a are located in the physical bank of the number $\bar{a}$ %N (% = remainder of the Euclidean division), at the address a/N (/ = quotient of the Euclidean division).

In conclusion, poor utilization of the passband of a multi-bank memory results in a combination of two factors that can be influenced only in an incidental manner. Unless only a few, standard access sequences are used, physical bank conflicts prove to be inevitable (and network conflicts even more so), causing a delay in processing some requests. In addition, there are always dependencies between the requests presented to the memory system (a dependency between two requests means that the second cannot be sent to memory until the first has been accepted). The delays that occur in the processing of requests linked by dependencies therefore have an impact on the processor(s) concerned, which stop transmitting entirely until the conflicts are resolved. This results in under-utilization of channels, and thus in poor utilization of the passband of the memory system as a whole.

An object of the invention is to remedy these drawbacks of the known systems and methods for arbitrating requests and resolving conflicts related to access to independent-bank memories for computing machines.

To this end, it seemed worthwhile to make use of a known processor-memory connection structure which so far had been applied only to the case of physical banks having only one module.

More specifically, a particularly simple and attractive processor/memory connection structure is known that encompasses not only the full crossbar system but also the shared bus, which is based on the logical bank concept. This parameterizable model, called "ML"(the abbreviation for Module Lines), which has been studied relatively comprehensively by F.A. Briggs in the case of one-module physical banks (where there is no distinction between word and superword), is represented in Fig. 1. In this structure, a logical bank is constituted by the association of a plurality of physical banks around a line constituted by a control bus and two data buses reserved respectively for writing and reading superwords; sharing of the line is warranted by the fact that each request mobilizes a bus for no more than one cycle, whereas the physical banks remain limited in rate to one read or one write of a superword per Tc cycles; the term "logical bank" derives from the fact that as long as there is good address distribution, the physical banks act as pipelined memories capable of accepting one request per cycle and servicing the requests with a latency Ta (Ta = time needed to access a module; for a static memory, Ta = Tc, while for dynamic memories, Tc = 1.3 Ta). In such an ML structure, the lines are connected in turn to channels via a crossbar network; thus, the access path to a logical bank is shared by all the processors. This scheme differs slightly from the memory system organization of a number of known machines, in which each section offers an access line reserved for each processor (in the case of a monoprocessor, a section is the same as a logical bank).

Hence, the invention concerns a system for arbitrating requests and resolving conflicts related to access to the independent-bank memories of computing machines of the type comprising at least one processor and a shared memory including at least one

logical bank constituted in turn by at least one physical bank, linked by independent access channels adapted to operate simultaneously to transfer up to one data item per clock cycle, said channels each comprising a write bus, a read bus and a control bus supplying in particular a priority data item issued by the processor, said system being characterized in that it comprises a controller associated with each logical bank to arbitrate conflicts between the requests for access to that logical bank and one freeze instruction cable per processor to deliver a freeze instruction signal designed to interrupt the issuing of requests by that processor, in that said controller includes, for each channel, an interface comprising write and control input buffers and read output buffers, connected by buses to the channel and to the logical bank between which they are inserted, a write/read manager connected to the buffers to manage the transfers between them and, on the one hand, the channels, and on the other hand, the logical bank, to the control bus, and to the freeze instruction cable, and a decoder connected to the control bus and to the write/read manager to indicate to the latter whether requests are intended for the logical bank, and, associated with said interface, which generates an identification tag for each request intended for it, an arbiter connected to the write/read managers and to the freeze instruction cable(s) to select from the requests the one that will access the logical bank and, if at least one request having an absolute priority is not satisfied, to order a freeze of the processor concerned, and a delay unit associated with the logical bank and connected to the write/read managers to propagate control data.

The invention also concerns a method for arbitrating requests and resolving conflicts related to access to independent-bank memories of computing machines of the type comprising at least one processor and a shared memory including at least one logical bank constituted in turn by at least one physical bank, linked by independent access channels adapted to operate simultaneously to transfer up to one data item per clock cycle, said channels each comprising a write bus, a read bus and a control bus supplying in particular an initial priority data item issued by the processor and constituted by the number of cycles for which the request can be delayed, said method being characterized in that for each logical bank, on each arrival of a request via a channel it is determined

whether the request is intended for said logical bank, and if so, the parameters of that request are stored temporarily; on each cycle, from all the requests that have previously been stored and have not yet been granted access to the physical bank(s) a request is chosen that can be passed on immediately without causing a conflict and is given access to the physical bank, and in the absence of an unsatisfied request having an absolute priority, the priority of all the other still-pending requests is decremented, whereas in the presence of at least one unsatisfied request having an absolute priority, access to the physical bank is given to a request that can be passed on immediately without causing a conflict and a temporary freeze signal is sent via the freeze instruction cable(s) connected to the processor(s) having unsatisfied requests having an absolute priority so as to interrupt the issuing of new requests by said processor(s).

By virtue of these characteristics, the goal of more or less completely eliminating bank or network conflicts, which proves to be unrealistic outside the domain of specialized applications, can be replaced by that of equitably distributing requests throughout the banks and lines, considering durations of a few cycles; this last criterion is easier to implement, since there are simple and effective hardware functions that can be used to satisfactorily uniformize the distribution among the modules of the addresses output by the processors. In addition, local accumulations of outstanding requests can then be absorbed through the suitable use of buffer stages as long as there is some independence among the requests presented to the memory system; in the past, the latter condition has been met only in the case of multiprocessors, where the inhibition of one processor does not affect the operation of the others; the concept of request priority set forth here represents a considerable advance, in that by making it possible to specify the degree of independence of the requests, it allows the advantages of using buffers with each processor to be exploited to an extent heretofore unknown.

Thus, the architectural principles applied to current supercomputers are brought to bear by the system and method for arbitrating requests and resolving conflicts according to the invention, which are particularly suitable for multi-bank and multi-access

memories intended for high-performance, parallel machines such as mini-supercomputers and image synthesizers.

Other characteristics and advantages of the invention will emerge from the following description of a preferred form and a preferred embodiment of the invention, provided by way of nonrestrictive example and illustrated in the appended drawings, wherein:

- Fig. 1 shows a block diagram of a known ML structure,

- Fig. 2 is a general block diagram of a system for arbitrating requests and resolving conflicts according to the invention,

- Fig. 3 is a block diagram of an arbiter equipping the arbitration system of Fig. 2,

- Fig. 4 is a time diagram for a channel management stage of the arbiter of Fig. 3,

- Fig. 5 is a block diagram showing in detail the circuits of a selector of the arbiter of Fig. 3,

Fig. 6 is a block diagram showing in detail the circuits of part of an interface block of the system from Fig. 2.

As has been seen, the basic ML processor/memory connection structure studied by Briggs (Fig. 1) is designed to connect p memory channels C to logical banks BL each constituted by N physical banks BP. The connection is made across controllers CO, by means of taps D and lines L in a ratio of one controller CO per logical bank BL. Each controller CO is therefore connected to the channels C by as many respective taps D as there are channels, and to each physical bank BP of the logical bank BL assigned to it via a line L that has as many branches as there are physical banks to be served.

In the system according to the invention, only part of which is illustrated in Fig. 2, the structure is similar to this basic ML structure and is adapted to the case in which a word and a superword are not the same, a superword being composed of eight bytes in the chosen design; thus, a physical bank is composed of eight modules of one byte each, each of them served by one sub-branch. The part of the inventive system depicted in Fig. 2 is that relating to the connection of the p channels C_0, $C_1 \ldots C_{p-1}$ emanating from the various processors to a single logical bank BLO composed of N physical banks

BPO_0, $BPO_1 \ldots BPO_{N-1}$, each corresponding to one superword composed of eight one-byte words; according to the invention, all the logical banks such as BLO are connected to p channels in the same manner, across a controller specific to each logical bank; the channels C_0, $C_1 \ldots C_{p-1}$ and their respective taps to the controller, as branches of the lines to each physical bank and the sub-branches, each comprise three buses, i.e. a control bus BUC, a write bus BUE and a read bus BUL. In addition to channels such as C_0, the exchange line between the processors and the memory includes a bus constituted by temporary-freeze instruction cables for ordering a temporary freeze of the processors CGP, and that bus serves all the controllers. There are as many freeze instruction cables as there are processors, although only one cable is illustrated in Fig. 2 to avoid complicating the drawing.

The controllers are all constituted in the same fashion, and the controller associated with one logical bank BLO will therefore be described more precisely. This controller chiefly comprises a single arbiter ARO and a single delay unit REO, and a number of other stages whose redundancy is equal to the number of channels C_0, $C_1 \ldots C_{p-1}$ emanating from the processors, each channel being assigned exclusively to one processor. These other stages are double buffers TCO_0, $TCO_1 \ldots TCO_{p-1}$, decoders DEO_0, $DEO_1 \ldots DEO_{p-1}$ and read/write management stages GEO_0, $GEO_1 \ldots GEO_{p-1}$. The buffers TCO_0, $TCO_1 \ldots TCO_{p-1}$ are each constituted by a (control and write) input buffer and a (read) output buffer, each connected (via an input and an input and via an output, respectively) to the bus corresponding to the channel assigned to them, by means of an aforesaid tap bus, and (via an output and an output and via an input, respectively) to the corresponding bus of the access line to the logical bank. Decoders DEO_0, $DEO_1 \ldots DEO_{p-1}$ each have their input connected to the control bus of the tap. Read/write management stages GEO_0, $GEO_1 \ldots GEO_{p-1}$ each have an input connected to the output of the corresponding decoder, an input connected to the control bus of the same tap, an input connected to the freeze instruction cable CGP for the processor concerned, which is also connected to one of the $\overline{GEL}$ [FREEZE] inputs/outputs of the common arbiter ARO for all the channels assigned to the same processor and delivering GEL (or more

precisely, $\overline{GEL}$) signals, an input connected to a corresponding output EXS of the arbiter, an input connected to the output of the delay unit REO, which, like the arbiter, is common to all the management stages of the logical bank, and whose input is connected to an output of the management stage, and an output connected to inputs ENE, ETE, PIE, ADE, PAE of the arbiter; each buffer stage is also connected to its respective management stage.

The management of such a priority-arbitrated ML memory system is therefore fully distributed at the logical bank level, the shared data being limited to the status of the freeze instruction lines connected to the cable CGP, and the functions of the various stages of the controller are distributed as follows:

- Each interface block (there is one per channel), comprising an input buffer, an output buffer, a management stage for writes to the input buffer / reads from the output buffer, and a decoder, is responsible for intercepting on the channel the requests intended for the logical bank it belongs to, and generates the tag used by the arbiter to identify each request.
- The arbiter per se, apart from the previously described functions of priority management and request selection, supplies the input buffers with the read addresses from its ETS output and the read validation signals from its EXS output upon transmission of the parameters of the selected request to the physical banks.
- The delay unit propagates control data relating to the write addresses and to the validation signals used to store the data returned by memory reads in the output buffers.

The inventive controller implementation procedure that has just been described is based on a concept that can be defined as the "initial priority" of a request, representing the maximum number of cycles for which the processing of the request can be delayed by the memory system without a temporary freeze being placed on the processor. This initial priority is a control data item that is to be transmitted by the processor in the same way as the address or the read/write signal; it is transmitted by the control bus BUC. During a read, the data item being output by the physical banks is rerouted to the output buffer associated with the channel it is coming from, and is entered there at an address

that is equal to the sum of the arrival date and the initial priority of its request (modulo a constant d); the read address of the output buffer is maintained by a modulo d counter that is incremented at the end of each cycle in the absence of a processor freeze. In this manner, the data item corresponding to a read arrives at the processor exactly (Min_L + initial priority) cycles after the issuing of the request concerned, regardless of the exact date when it was passed to the physical banks and the number of freezes that have intervened. The number Min_L represents the minimum latency of the memory system (a priori Tc + 3), and the cycles are counted from the viewpoint of the processor, which, by definition, does not "see" the freezes.

The "current" priority of a request corresponds to its initial priority minus the number of cycles that have elapsed since it was issued by a processor, not counting the cycles during which that processor was frozen.

As a result of this implementation, in practice only a reduced fraction of the requests call for a response time close to the minimum permitted by the memory system. For example, during the loading of an instruction buffer or a vectorial register, only the first words are urgent (initial priority close to zero), since the data are used sequentially. Similarly, in reading a cache line, the word that is the source of the error is more urgent than the others. On the other hand, all the requests coming from an input/output controller can accept a high initial priority, and the same is true of most memory writes resulting from the execution of a globally optimized code.

Assuming that there is only one processor, the procedure for implementing the system that has been described is as follows: the p channels transmit a total of between zero and p requests every cycle. At the level of each logical bank, the valid addresses on the channels are decoded to determine whether the request is intended for the logical bank in question. If so, the parameters of the request are stored in the channel-dedicated input buffer registers in the logical bank, pending transmission to the physical banks. The arbitration circuit ARO of the logical bank is also informed that one or more new requests are coming in, and stores the following attributes of those requests:

- a tag that represents the address of the request attributes in the input buffer,

- an initial priority: the number of cycles for which the request can be delayed,
- data that include the number of the physical bank to which the request pertains,
- a description of the words accessed within the superword.

Simultaneously, the arbiter ARO selects from the previously queued requests the one that will access memory. For this purpose, it chooses the request having the lowest current priority from among those that can be passed on immediately without causing a physical bank conflict. In the absence of a freeze instruction, the current priorities of all the pending requests in the L arbiters of the memory system are decremented and a new cycle begins. If a given arbiter contains a request that has an absolute priority, for example zero, and has not passed to memory, a processor freeze is ordered over the wired OR line connecting the arbiter ARO to the freeze cable CGP, and none of the arbiters decrements the current priorities of its requests. During a processor freeze the arrival of new requests is stopped and the priorities are no longer decremented, but the arbiters of the various logical banks continue to stream the previously queued results to the physical banks of their logical bank in normal fashion.

The system and method that have just been described can be applied to a large number of cases, and in particular, with equal validity, to a multiprocessor and a monoprocessor. In the case of specific application to a multiprocessor, as has been seen, on the one hand it must be assumed that each channel is reserved for one of the processors, and on the other hand, one freeze line CGP must be provided per processor, the reason being that in a multiprocessor, when a request having an absolute priority of zero comes in and is unable to access memory, only the processor from which it originated need be temporarily frozen. Under these conditions, the invention can for example be applied to memories having characteristics similar to those of existing computers, for example composed of $N = 64$ physical banks of cycle time $Tc = 4$ with interleaving on the lower portions of the addresses, the number of channels connected to that memory being $p = 8$, for a ratio $N + Tc$ equal to $2p$; given these 64 physical banks, and since, on the one hand, according to Briggs the number of lines L must be greater than p, which equals 8 in the system under consideration, and on the other hand, the

cycle time of a physical bank is equal to 4, it does not seem very worthwhile to provide less than four of them per logical bank, i.e. L less than or equal to 16; thus, taking L as a power of two, the only possible choice is the number sixteen.

With regard to the allocation of physical banks to logical banks, physical banks 0, 1, 2, 3 can be assigned to logical bank 0, physical banks 4, 5, 6, 7 to logical bank 1, ... physical banks 28, 29, 30, 31 to logical bank 15; or, alternatively, physical banks 0, 16, 32, 48 to logical bank 0, physical banks 1, 17, 33, 49 to logical bank 2, ... physical banks 15, 31, 47, 63 to logical bank 15. The latter scheme theoretically offers the best performance, since the requests are distributed more uniformly through all the logical banks in the important case of address increments of a superword; if this system has not been appropriated for some known machines, this is due to the occurrence of "linked conflicts" resulting from a number of lines equal to the cycle time of the physical banks; since the inventive priority system has been designed to absorb that type of problem, this distribution can be used.

Experience therefore shows that if a memory such as that defined hereinabove is exposed to a stream of requests randomly distributed in the addressing space and arriving at the maximum permitted rate, assuming eight channels, for example, the results are not significantly affected by changes in the assignment of the physical banks to the logical banks. Moreover, once there are more than eight initial priorities the results are very similar in the case of an octoprocessor with one channel per processor, a quadriprocessor with two channels per processor, a biprocessor with four channels per processor and a monoprocessor with eight channels. A synchronous monoprocessor with eight channels yields better results at an initial priority of 4 or more than an octoprocessor with 0 and 1 degrees of buffering, and in addition all the machines perform well, with a 5% variation, beyond an initial priority of 12; this means that by virtue of the invention, the multiprocessor option is no longer an obligation for the parallel-machine designer once request processing latency becomes secondary due to the asymptotic processing rate.

The latter condition is verified particularly in the context of vectorial calculus.

Thus, the invention also has a particularly attractive application in the case of an eight-channel monoprocessor in which the request addresses are evenly spaced in memory; in this case in vectorial calculus, around 90% of accesses correspond to such streams of evenly spaced requests separated by an increment of s superwords, with $s = 1$ alone accounting for about 70%. In this situation, it is primarily the total number n(s) of physical banks scanned by the request stream that determines the efficiency of the memory system; to prevent the catastrophic performance degradations that affect all memory systems of supercomputers using increments s that are multiples of 4, the supercomputer can be equipped with a "biasing" system that carries out a particular transformation for each address *a* issued by the processor; such a system, based on the use of a hardware operator performing fast Euclidean division by a prime number, makes it possible to keep problems of poor address distribution to a rarity, and in the present case does not require any other adaptation of the logical bank/physical bank distribution or any change in the request arbitration mechanism. Under these circumstances, conflicts occur only if two or more requests simultaneously ask for one line (network conflict) or if, with certain increments, there is a local accumulation of requests pertaining to the same physical bank; the effect of these conflicts disappears as soon as the memory system is allowed some freedom in processing the requests (in this example, initial priorities of 6 or above). It will be appreciated that the fundamental problems entailed by having a microprocessor use the passband of a central memory equipped with independent banks can therefore be considered to be solved satisfactorily by the combination of an ML structure, this priority management system and a simple address biasing function, which in the final analysis eliminates the technological reasons for creating a multiprocessor, since a simpler and higher-performance system is obtained in this way.

It is also possible to envision a specific machine that would take full advantage of the system and method according to the invention.

A preferred form of the invention that can additionally comprise some variants with respect to the "model" described hereinabove will therefore now be described for

particular application to a machine having a theoretical power of 160 MIPS or 80 MFLOPS on 64 bits for a cycle time of 50 ns, based on current TTL and CMOS technology. The machine in question comprises a processing unit with two pipelined floating multipliers, two pipelined floating arithmetical logic units, four address operators and four independent memory channels (p = 4). The memory system of this machine comprises N = 32 physical banks of 8 bytes each, grouped together as L = 8 lines of M = 4 modules. The occupation time Tc of the physical banks is two cycles (less than 200 ns). The initial priorities range from 1 to 15 (d = 16) for a minimum response time of Tc + 3 cycles.

The architecture of this machine distinguishes between eight types of data: signed bytes (8 bits), unsigned bytes (8 bits), signed short integers (16 bits), unsigned short integers (16 bits), long integers (32 bits), single precision floating points (32 bits), double precision floating points (64 bits), and finally, logic values (64 bits). A word therefore corresponds to one byte and a superword to 8 bytes or 64 bits. According to the alignment constraints, any data item that is *l* bits in size must reside at an address that is a multiple of 2^{l-3}. The concept of a hyperword is also introduced, which is a data item of *L* consecutive superwords with each superword residing in a separate logical bank. The reading or writing of hyperwords is performed in exactly the same manner as simple read or write operations, except that all the logical banks are selected when a valid address appears on a channel.

The foregoing exemplary embodiment of the invention is based solely on current TTL or CMOS technology, as has been seen, and a direct extension of performance can be obtained merely by changing technologies (for example, switching to the ECL technique and BI-CMOS).

The portion of the inventive control system that will more precisely be the subject of the following description and of Fig. 3 is the arbiter (for example ARO) of that system. To begin with, it should be noted that the clock is designed to be able to support the use of dynamic logic on the critical paths. To this end, the clock delivers two frequency-squared 20 MHz signals offset by one quarter period (12.5 ns), hereinafter denoted

MCK and SCK; the time origin is set upon stabilization of the $\overline{GEL}$ signal, during the rising edge of MCK.

This arbiter must cycle in less than 50 ns, and that period has to include the distribution/sampling time of the processor freeze instruction, or about ten nanoseconds via coaxial cable. The arbiter is therefore pipelined, with the new-request insertion phase overlapping with the selection phase for access to the physical bank. As a result, initial priorities of 0 are not accepted even though current priorities may reach that value internally. The arbitration circuit, illustrated in a general manner in Fig. 3, is constituted by three types of blocks, i.e. four sequencers ORO_0, ORO_1...ORO_3, which correspond respectively to the four channels of the machine and whose functions are to store the incoming requests in the classes corresponding to their initial priority, to search for the request having the lowest current priority, and to update the priorities according to the status of the GEL pin; four estimators ESO_0, ESO_1...ESO_3, responsible for determining whether the requests proposed by the sequencers can be passed to memory without causing any conflict and which contain a representation of the current occupation status of the banks, the updates being performed at the end of each cycle to take into account the passage of time and any transmission of a new request to a physical bank; and a selector ELO, which collects the requests proposed by the sequencers after they have passed through the estimator, chooses the non-conflicting request having the lowest current priority and resends its characteristics to the estimators so the internal states can be updated, and also activates the freeze instruction when there are still requests with a current priority of 0 that cannot be passed to memory.

More precisely, all the inputs of the arbiter correspond to inputs of the sequencers, such that the data from each channel are transferred directly to the corresponding sequencer.

Thus, each sequencer comprises:

- a validation input ENE connected to the corresponding read/write management stage, for receiving therefrom a validation signal indicating that the request should be recorded;

- an identification tag input ETE, also connected to the management stage, for receiving therefrom a four-bit identification tag signal of between 0 and 15 (0 and d –1),
- an initial priority input PIE connected to the management stage, for receiving therefrom a four-bit initial priority signal of between 1 and 15 (1 to d – 1),
- a physical bank number input ADE connected to the management stage, for receiving therefrom a signal representing the number of the physical bank to which the incoming request pertains,
- a description input PAE connected to the management stage, for receiving therefrom an encoded description signal describing the bytes accessed in the physical bank.

In the present case, only two bits are needed for the physical bank number and four bits for the description of the bytes accessed in the bank, since there are only fifteen possibilities for the occupation of the bytes in a 64-bit superword if the read is limited to words of 8, 16, 32 and 64 bits aligned with addresses that are respectively multiples of 1, 2, 4 and 8.

The sequencers, on the other hand, have two other inputs which, instead of being connected to an output of a channel interface, are connected to an output of the selector associated with the sequencer, as regards the first input, and to the system freeze line, as regards the second. Thus, the sequencers comprise:
- an extraction input EXE receiving an extraction signal indicating that the request proposed in the preceding cycle is being passed to memory and that all the data relating to it can be retrieved;
- an enable decrementation input PUE to enable decrementation of all the priorities of pending requests, since a freeze did not occur.

Apart from the tag output ETS, which, as has been seen, is connected to the corresponding read/write management stage, the outputs of the sequencers are all reserved for internal use by the arbiter. They are used either directly by the corresponding estimator and the selector or indirectly by the selector after being pooled with the estimator output. Thus, the sequencers also comprise:

- a current priority indication output PRS, whose output signal indicates the current priority of the proposed request, this data item of between 0 and 14 (0 and d – 2) being used only in the selection process,
- an output USS juxtaposing the "fields" of the description signal for the accessed bytes and the physical bank number signal; the associated six-bit output signal, corresponding to the request proposed by the sequencer, is applied directly to the estimator concerned and to the selector respectively via identification inputs IDE and USE of these elements, and is stored for later use in the selector for one-half cycle,
- a vacancy indication output VAS, supplying a vacancy signal indicating whether the sequencer is empty so that the signals from the preceding outputs ETS, PRS and USS are not taken into account,
- a freeze instruction output DGS, serving to supply a signal intended for the selector and formulating a freeze instruction by setting to a "1" state when there is more than one zero-priority request in the sequencer,
- an accelerate freeze output AGS to supply a "1" state signal accelerating the calculation of the freeze request when a valid request of priority 0 is presented by the sequencer; the state of this signal is determined simply according to the state of the output signals at the current priority indication output PRS and the vacancy indication output VAS.

In addition to the previously mentioned six-bit identification input IDE, which uses the aforesaid juxtaposition output signal to identify the bytes accessed in the physical bank and the number of that physical bank for the request proposed by the sequencer, the estimators comprise, on the one hand:
- a parameterization input COE supplying the estimator with a parameterization signal that tells it the occupation time in cycles of the modules, the state of the input pins COE being hardwired,
- a conflict found output CCS delivering a signal positioned at "1" when the request presented via conflict identification input IDE cannot be passed to memory without causing a conflict.

The estimators also comprise, on the other hand, an update input MJE for receiving a signal updating their representation of the occupation status of the physical banks, said input being connected for this purpose to an output MJS of the arbiter.

Finally, they comprise a mode switching input ECE connected to receive the MCK'.S CK' signal.

The estimators consequently operate in two modes switched by switching input ECE; during the first phase, the characteristics of the request proposed by the sequencer are used to detect whether a module conflict is likely to occur, and the signals present at identification input IDE, parameterization input COS, and conflict found output CCS are used; during the second phase, the characteristics of the request chosen by the selector for passage to memory are presented to all the estimators at their respective update inputs MJE so they can update their representation of the occupation status of the physical banks.

Under these conditions the circuits of the arbiter require twenty input and/or output pins per channel, to which number must be added:

- a freeze terminal GEL (output of the selector) for the processor freeze circuits, connected to the decrementation inputs PUE of each estimator and, as has been seen, to the processor freeze instruction cable CGP, supplying a signal $\overline{GEL}$,
- a reset-to-zero terminal RAZ, not shown in Fig. 3, for reset-to-zero circuits generating a reset-to-zero signal RZ,
- two terminals not shown, for the clock signals MCK and SCK supplied in particular to the estimators and the sequencers by a clock also not shown,
- four terminals not shown, for the power supplies.

The selector also comprises extraction output terminals EXS for supplying the respective extraction signals to the sequencers and as output from the arbiter, as previously mentioned.

The selector also comprises current priority indication inputs PRE respectively connected to the current priority indication outputs PRS of the sequencers, and inputs AGE connected to the accelerate freeze outputs AGS of the sequencers. It also

comprises request-validity and request-interrupt inputs CEE and HRE, respectively, connected to the vacancy VAS, freeze instruction DGS and request acceleration AGS outputs of the sequencers, and to the conflict found outputs CCS of the estimators by means of gates P_1, P_2, P_3: there is, naturally, one pair of these inputs per sequencer/estimator group, there being three gates per sequencer/estimator group: an AND gate P_1 has an input connected to request acceleration output AGS and an input connected to conflict found output CCS, while its output is connected to an input of an OR gate P_2, another input of which is connected to freeze instruction output DGS and whose output is connected to interrupt input HRE of the selector, OR gate P_3 having an input connected to conflict found output CCS and an input connected to vacancy output VAS, while its output is connected to request-validity input CEE of the selector.

The arbiter is therefore constituted by management circuits, each corresponding to a channel and each comprising a sequencer, an estimator, and gates serving to shape certain signals intended for the selector and optionally integrated into a single circuit. This configuration makes it possible to escape the limitations of the VLSI technology used, input multiplexing not being conceivable otherwise for reasons of performance; it also makes it possible not to limit the design a priori to a given number of channels.

The time diagram of a channel manager according to the configuration that has just been described is depicted in Fig. 4, in which added to the first and second clock signals MCK and SCK are third and fourth clock signals MC[K]' and SCK' corresponding to them, with a delay of 7 ns after input/amplification in the circuit; as a general rule, a signal takes 4 ns to make its way into the manager and 10 ns to leave it; these clock signals are approximately trapezoidal; for convenience, the signals appearing at the inputs and outputs of the various circuits are denoted by the first two letters of the reference for the input or output at which they appear; thus, it can be seen that decrementation signal PU and initial priority signal PI appearing at inputs PUE and PIE of the sequencer are valid from the rising edge of the first clock signal MCK to the rising edge of the third clock signal MCK', whereas validation signal EN, identification tag signal ET, physical bank number signal AD and description signal PA, appearing at

the respective inputs ENE, ETE, ADE, PAE of the sequencer, are valid from the falling edge of the first clock signal to the falling edge of the third, and the extraction signal EX that appears at input EXE of the sequencer during the rising edge of the first is valid until the rising edge of the third.

Reading on the part of the sequencer occurs when the third clock signal MCK' is at the high level, and writing between the falling edge of the fourth clock signal SCK' and the rising edge of the third; reading by the estimator begins between the rising edge of the fourth and the falling edge of the third, to terminate during the falling edge of the aforesaid fourth; writing by the estimator takes place between the rising edges of the third and fourth clock signals MCK', SCK'.

The freeze request acceleration signal AG at the output of the sequencer does not stabilize much after the rising edge of the fourth clock signal SCK', while the same is true of tag signal ET, the signals juxtaposing the fields of the description and physical bank number signals US and the current priority indication signal PR, also at the sequencer output, slightly before the falling edge of the third clock signal MCK', and is also true of the request-interrupt HR and priority-validity CE signals at the selector input between the falling edges of the third and fourth clock signals MCK', SCK'; all these signals remain stable at least until the rising edge of the fourth clock signal SCK'.
xx In the arbiter so managed, the selector circuit, which is the only part built from standard components, is therefore intended to perform, on the one hand, the selection of the request intended for memory, and on the other hand, the calculation of the freeze instruction at the logical bank level.

As mentioned above, a freeze is ordered at the logical bank level if at least one of the sequencers contains a valid zero-priority request that has not been passed to memory. By virtue of the pipelined operation of the arbiter, this calculation is done in advance so that the freeze instruction does not come too late. The freeze condition can thus be broken down into three parts, the first bringing about the instructed advance timing:

- If one of the sequencers has a plurality of zero-priority requests (signal DG at one), a freeze will definitely have to be triggered for the next cycle.
- If a plurality of zero-priority requests are simultaneously proposed to the selector, no more than one will be passed to memory, so a freeze must be expected here as well.
- If a zero-priority request is declared to be in potential conflict by the estimator a freeze must occur, since that request will be delayed.

The freeze instruction signal GR is the equivalent of disjunction of the three preceding conditions, which for a four-channel memory system is written: $GR = DG_0 + DG_1 + DG_2 + DG_3 + AG_0 \cdot AG1 + AG_0 \cdot AG_2 + AG_0 \cdot AG_3 + AG_1 \cdot AG_2 + AG_1 \cdot AG_3 + AG_2 \cdot AG_3 + AG_3 + AG_0 \cdot CC_0 + AG_1 \cdot CC_1 + AG_2 \cdot CC_2 + AG_3 \cdot CC_3$

This is because the symmetric polynomial σ_k in n variables equals 1 in Boolean algebra when at least k of these variables are not zero (k = 2 here). Denoting by HR_i the signal equal to $DG_i + AG_i \cdot CC_0$, we obtain:

$GR = HR_0 + HR_1 + HR_2 + HR_3 + AG_0 \cdot AG_1 + AG_0 \cdot AG_2 + AG_0 \cdot AG_3 + AG_1 \cdot AG_2 + AG_1 \cdot AG_3 + AG_2 \cdot AG_3$

The selection of the request for access to memory is performed in a simple manner, since each channel manager proposes a request whose current priority is indicated by the field PR and validity by CE. The latter signal is the logical OR of the VA signal, which indicates whether the sequencer is empty, and the CC signal triggered by the estimator when a proposed request is in collision with the current occupation of the modules. The request that is to be selected is the one of the valid requests that has the lowest priority. The EX signal, which tells each sequencer that its request has been chosen, is therefore written (example for channel 0):

$EX_0 = \overline{CE}_0 \cdot ((CE_0 < CE_1) + (CE_0 = CE_1) \cdot (PR_0 < PR_1)) \cdot ((CE_0 = CE_2) + (CE_0 = CE_2) \cdot (PR_0 < PR_2)) \cdot ((CE_0 < CE_3) + (CE_0 = CE_3) \cdot (PR_0 < PR_3))$

Knowing that $(CE_0 < C_1)$ is written for Boolean values $\overline{CE}_0 \cdot CE_1$, and furthermore that $(CE_0 = CE_1)$ is equivalent to $CE_0 \cdot CE_1 + \overline{CE}_0 \cdot \overline{CE}_1$, we obtain after simplification:

$EX_0 = \overline{CE}_0 \cdot (CE_1 + (PR_0 < PR_1)) \cdot (CE_2 + PR_0 < PR_2) \cdot (CE_3 + (PR_0 < PR_3))$

$(PR_i < PR_j)$ is calculated by collecting the output carry from the operation $PR_j - PR_i$ as a 1's complement, i.e. actually $\overline{PR}_i + PRj$. This operation is performed here by means of standard TTL adders of series AS181 or AS881. As a result of this choice, it is further advantageously possible to install a circular precedence mechanism permitting an equitable choice among the channels when two or more valid requests having the same priority are proposed simultaneously. This is because, as will be observed from the expression for EX_i, in the latter case none of the requests is valid owing to the strict nature of the inequalities. The solution is to transform some evaluations of $(PR_j < PR_i)$ into $PR_i \leq PR_j)$ using the following elementary property: $(PR_i \leq PR_j) <=> (\overline{PR}_i + PR_j + 1) \geq 2^4$. The additional 1-term is introduced into the 'AS881 [sic] adders as an input carry, and the result of the comparison is given by the value of the output carry.

The circular precedence among the four channels has four periods. During the first, channel 0 has a higher priority than channel 1, which in turn has a higher priority than channel 2, etc. During the second, a circular permutation over the precedences makes channel 1 the highest-priority channel, followed by channel 2, ... down to channel 0. Thus:

T0: $EX_0 = \overline{CE}_0 \cdot (CE_1 + (PR_0 \leq PR_1)) \cdot (CE_2 + (PR_0 \leq PR_2)) \cdot (CE_3 + (PR_0 \leq PR_3))$

$EX_1 = \overline{CE}_1 \cdot (CE_2 + (PR_1 \leq PR_2)) \cdot (CE_3 + (PR_1 \leq PR_3)) \cdot (CE_0 + (PR_1 < PR_0))$

$EX_2 = \overline{CE}_2 \cdot (CE_3 + (PR_2 \leq PR_3)) \cdot (CE_0 + (PR_2 < PR_0)) \cdot (CE_1 + (PR_2 < PR_1))$

$EX_3 = \overline{CE}_3 \cdot (CE_0 + (PR_3 < PR_0)) \cdot (CE_2 + (PR_3 < PR_1)) \cdot (CE_2 + (PR_3 < PR_2))$

T1: $EX_0 = \overline{CE}_0 \cdot (CE_1 + (PR_0 < PR_1)) \cdot (CE_2 + (PR_0 < PR_2)) \cdot (CE_3 + (PR_0 < PR_3))$

$EX_1 = \overline{CE}_1 \cdot (CE_2 + (PR_1 \leq PR_2)) \cdot (CE_3 + (PR_1 \leq PR_3)) \cdot (CE_0 + (PR_1 \leq PR_0))$

$EX_2 = \overline{CE}_2 \cdot (CE_3 + (PR_2 \leq PR_3)) \cdot (CE_0 + (PR_2 \leq PR_0)) \cdot (CE_1 + (PR_2 < PR_1))$

$EX_3 = \overline{CE}_3 \cdot (CE_0 + (PR_3 \leq PR_0)) \cdot (CE_1 + (PR_3 < PR_1)) \cdot (CE_2 + (PR_3 < PR_2))$

T2: $EX_0 = \overline{CE}_0 \cdot (CE_1 + (PR_0 \leq PR_1)) \cdot (CE_2 + (PR_0 < PR_2)) \cdot (CE_3 + (PR_0 < PR_3))$

$EX_1 = \overline{CE}_1 \cdot (CE_2 + (PR_1 < PR_2)) \cdot (CE_3 + (PR_1 < PR_3)) \cdot (CE_0 + (PR_1 < PR_0))$

$$EX_2 = \overline{CE}_2 \cdot (CE_3 + (PR_2 \leq PR_3)) \cdot (CE_0 + (PR_2 \leq PR_0)) \cdot (CE_1 + (PR_2 \leq PR_1))$$

$$EX_3 = \overline{CE}_3 \cdot (CE_0 + (PR_3 \leq PR_0)) \cdot (CE_1 + (PR_3 \leq PR_1)) \cdot (CE_2 + (PR_3 < PR_2))$$

T3: $$EX_0 = \overline{CE}_0 \cdot (CE_1 + (PR_0 \leq PR_1)) \cdot (CE_2 + (PR_0 \leq PR_2)) \cdot (CE_3 + (PR_0 < PR_3))$$

$$EX_1 = \overline{CE}_1 \cdot (CE_2 + (PR_1 \leq PR_2)) \cdot (CE_3 + (PR_1 < PR_3)) \cdot (CE_0 + (PR_1 < PR_0))$$

$$EX_2 = \overline{CE}_2 \cdot (CE_3 + (PR_2 < PR_3)) \cdot (CE_0 + (PR_2 < PR_0)) \cdot (CE_1 + (PR_2 < PR_1))$$

$$EX_3 = \overline{CE}_3 \cdot (CE_0 + (PR_3 \leq PR_0)) \cdot (CE_1 + (PR_3 \leq PR_1)) \cdot (CE_2 + (PR_3 \leq PR_2))$$

This mechanism is created by generating the adder input carries by means of properly initialized circular counters, and can be generalized to any number of channels without difficulty. A fixed precedence among the channels would be less equitable, although it would have the advantage of ensuring reproducible situations.

The arbiter depicted in Fig. 5 is adapted to this mode of operation, and each of the channel managers as defined hereinabove (i.e., each constituted by a sequencer, an estimator and gates) is illustrated therein as a single block GCO_0, GCO_1...GCO_3 equipped only with its inputs and outputs that are not the inputs and outputs interconnecting its component parts or the inputs from the associated read/write management stages GEO_0, GEO_1...GEO_3. Each channel manager is therefore depicted with the following inputs: extraction EXE (for the extraction signal EX, or more precisely $\overline{EX}$), enable decrementation PUE, update MJE, and reset RAZ (receiving a reset signal RZ); and the following outputs: current priority indication PRS, description field and physical bank number field juxtaposition USS, accelerate freeze request calculation AGS, interrupt HRS (delivering the interrupt signal HR, or more precisely $\overline{HR}$), and priority validity CES (delivering the priority validity signal CE, or more precisely $\overline{CE}$).

The channel managers have their accelerate freeze request outputs AGS connected in pairs to two respective inputs of an AND gate (six gates in all); the respective outputs of these AND gates are connected in pairs to two respective inputs of an $\overline{OR}$ gate (three gates in all); the outputs of the three $\overline{OR}$ gates and the request-interrupt outputs HRS of

the four channel managers are connected to seven respective inputs of a single $\overline{AND}$ gate whose output supplies the freeze instruction signal GR, which is applied to the input of an inverter constituted by an open-collector gate, which, creating a wired OR circuit, supplies as output the $\overline{GEL}$ signal applied on the one hand to the incrementation and decrementation inputs PUE of the channel managers and on the other hand to the input of a register constituted by a flipflop D synchronized by the second clock signal SCK, where it is stored to supply as output a local freeze signal $\overline{RG}$ for the interface blocks.

In terms of the selection logic, for each channel, three comparators are connected at one of their inputs to the current priority indication output PRS of the corresponding channel manager, and at the other input to the PRS output of another respective manager: the (inverting) carry inputs of the comparators are connected to the register outputs (flipflops D) of three respective counters, each comprising four registers designed to create the circular precedence mechanism; each of the (inverting) carry outputs drives one input of a respective AND gate whose other input is connected to the priority validity output CES of said other channel manager, whose PRS output is connected to the input of the corresponding comparator; the three respective outputs of the three AND gates corresponding to a single manager are connected to inputs of an $\overline{OR}$ gate and to the priority validity output of the same manager by appropriate circuitry.

The output of the $\overline{OR}$ gate (there is one per channel) is connected, on the one hand, to the extraction input EXE of the corresponding manager, and to the inverting control input of a three-state amplifier whose main input is connected to the description and physical bank number field juxtaposition output USS of the same manager and whose output is connected on the one hand to the update input MJE of all the managers[1] of the arbiter, which is further connected to circuitry the function of which will be stated below and which is known as a "pull-down." The output of the $\overline{OR}$ gate is also

---

1 TRANSLATOR'S NOTE: The French actually reads "questionnaires," which we take to be a typographical error for "gestionnaires" (managers).

connected to an input of a flipflop D synchronized by the first clock signal MCK, whose output supplies a signal $\overline{REX}$ that corresponds to the EX signal coming from the EXC output that can be seen in Fig. 2.

The counters have a ring structure and their flipflops are reset to zero by the RZ signal and synchronized by the second clock signal SCK.

Thus, in terms of the freeze control calculation logic, the accelerate freeze request signals AG drive the arrays of AND/$\overline{OR}$ gates, whose outputs are combined with the request-interrupt signals HR (or rather, $\overline{HR}$) by the seven-input $\overline{AND}$ gate; the resulting freeze instruction signal GR is then inverted by the open-collector gate, whereas the local freeze signal RG for the interface blocks is produced by storing the state of the freeze signal $\overline{GEL}$, as has been seen.

With regard to the selection logic, for each channel three comparisons are performed with the priorities proposed by the other managers, the input carries coming from the three counters and the output carries driving an AND/$\overline{OR}$ gate array that generates the extraction signal; the latter orders the description and physical bank number signal US of its channel to be transmitted over the input bus of the update signal MJE, and if no extraction signal is valid, the characteristics of a zero request are forced by the "pull-down," while as for the extraction-signal outputs intended for the interface blocks, these are stored by the aforesaid flipflop.

As has been seen, the arbiter thus is the crux of the inventive system, but said system cannot be limited to this arbiter, and interface blocks must be inserted on the path of the data between the channels and the modules; the composition of these interface blocks has already been sketched in very generally, and it has been noted that they are each constituted by buffers (an input buffer and an output buffer) and a mechanism for managing writes to the input buffer/reads from the output buffer, plus a decoder. In reality an interface block is necessarily more complex, and Fig. 6 provides a detailed representation of such an interface block.

Recognizable in Fig. 6 are a channel C_0, a logical bank BLO constituted by physical banks and the delay unit REO, and an arbiter constituted by a selector ELO and channel managers, only one manager GCO_0 being shown.

As has been seen, channel C_0 is connected to logical bank BLO on the one hand by means of a write and control bus which, more precisely, cascade-connects channel C_0 to a demultiplexer DM_1O, demultiplexer DM_1O to an input buffer TEO, input buffer TEO to a three-state amplifier SP_1O (controlled by the REX signal received from the selector ELO, and further receiving an ident signal designed to inform the logical bank of the channel that is the source of the request), and three-state amplifier SP_1O to logical bank BLO; logical bank BLO is further connected to channel C_0 by means of a read bus, passing through an output buffer TSO, an alignment/multiplexing logic LOO, and another three-state amplifier SP_2O.

In addition, demultiplexer DM_1O and input buffer TEO are connected by, on the one hand, a decoder DCO, and on the other hand, a combinatorial circuit PAO, using outputs PAS, LOS of demultiplexer DM_1O; these outputs are further connected respectively to the description input PAE and the physical bank number input ADE of channel manager GCO_0, and more precisely, of the sequencer.

Channel C_0 is also connected to the input of a demultiplexer DM_2O whose output is connected to the input of an AND gate whose output is connected to the input of a modulo d (= 16) counter CP_1O synchronized by the first clock signal MCK and whose output is connected, on the one hand, to the tag input ETE of manager GEO_0 and on the other hand to a write address input AEE of input buffer TEO; the output of the AND gate is also connected to the input of an inverter whose output is connected to a write validation input EVE of input buffer TEO and to the validation input ENE of channel manager GCO_0 to provide it with its validation signal $\overline{EN}$; in addition, input buffer TEO also has a read address input ALE connected to the tag output ETS of channel manager GCO_0 and a read/write control input LEE for receiving the second (read/$\overline{write}$) clock signal SCK.

Besides being connected by the read bus, which, like the write and control buses, need not be identified and described in detail, logical bank BLO and channel C_0 are connected by lines coming from the logical bank and driving the output buffer primarily via write address inputs AEE and write validation input EVE[2] for receiving the AE and $\overline{VE}$ signals therefrom.

Output buffer TSO is also synchronized, via a read/write control input LEE, by the first clock signal MCK or more precisely $\overline{MCK}$ (read/ $\overline{write}$ signal). It comprises read address inputs ALE connected on the one hand to outputs of a second modulo d counter (with d = 16) CP_2O whose input carries are supplied by the local freeze signal $\overline{RG}$. The outputs of the counter are also connected to inputs of a first adder whose other inputs are connected to the outputs of a second adder whose inputs for a first operand are connected to the priority output of demultiplexer DM_2O and whose inputs for a second operand are wired to represent a set number, for example Mem L + 2 according to the embodiment; the output of the first adder is connected to an input of input buffer TEO; the priority output of demultiplexer DM_2O is also connected to the inputs of decoder DEO_0, a decoder whose inverting outputs drive a system of parallel-loaded offset registers (flipflops D), by way of $\overline{AND}$ gates (except for the last one, which is driven by an inverter); the output of demultiplexer DM_2O is also connected to the initial priority input PIE of channel manager GCO_0.

The clock inputs of the offset registers are connected to the output of a logic system constituted by an $\overline{AND}$ gate and a cascaded inverter, the $\overline{AND}$ gate having one input fed by the first clock signal $\overline{MCK}$ and another input by the local freeze signal $\overline{RG}$. In addition, the parallel-loading control of the offset registers is connected to the output of an $\overline{AND}$ gate whose input receives the output signal of the AND gate disposed at the output of the second demultiplexer DM_2O, and another input receives the output signal of this demultiplexer signifying that the request under consideration

---

2 TRANSLATOR'S NOTE: Mistyped in the French text here as VEE.

is for a memory read; the output of the last flipflop of the offset register controls the second three-state amplifier SP_2O.

Thus, the channel interface blocks are located on the data path between the channels and the modules, making it possible to offer temporary storage capacity for the request attributes, thus bringing about a time disconnect between the date of issuance and the date of passage to memory that is essential to the efficiency of the priority management mechanism to which the invention is directed. The outputs of the two register banks constituting the two buffers are amplified to drive the bus, and the control signals (validation, address) are generated by this interface block with respect to writing to the input buffer (parameters of the requests) and reading from the output buffer (results of memory reads); on the other hand, reading of the input buffer is controlled by the arbiter (AND address and validation by REX), and control of writing to the output buffers emanates from the delay element associated with the physical banks.

When a request is intended for the logical bank to which the channel interface block belongs, the decoding logic validates the writing of the associated parameters into the input buffer and triggers recording of the attributes by the manager. The signal emanating from the AND gate mounted at the output of demultiplexer DM_2O is also injected as the input carry of the first modulo d (= 16) counter, which generates the write addresses in the input buffer, which are interpreted by the manager as a tag signal for the requests. The input buffer stores any data item that is to be recorded in memory; the address that is to be sent to the modules; a memory read/write signal (more precisely, read/$\overline{write}$); the physical bank address, fully decoded by the decoder DCO; the description signal PA, transcoded to byte occupation by the combinatorial circuit PAO; and finally, the write address in the output buffer of any data item returned from a memory read, supplied by the first adder.

When the request performs a memory read, the fact that a data item must be transmitted over the channel's return bus at the end of a cycle number equal to the initial priority plus Min_L (not counting freezes) is recorded in the parallel-loaded

offset register. In normal mode, this register offsets a control word whose output bit controls the read amplifier of the output buffer. The read address of the output buffer is supplied by the second modulo counter, incremented at each cycle in the absence of a freeze instruction. When a read request is decoded by the interface block, the associated control bit is input in the offset register by means of a device that performs the logical OR operation of the preceding control word and of the output of the decoder having d outputs that receives the value of the priority as input.

The write address in the output buffer is calculated by taking the current value of the counter that generates the read addresses and adding to it the priority of the request plus a constant. The value of the latter is to be adjusted as a function of the latency of the memory modules (around Mem_L + 2), as is the length of the portion of the offset register located between the decoder and the read amplifier.

The invention is, of course, not limited to the forms and embodiments described and illustrated hereinabove, and other forms and embodiments may be envisioned without departing from its ambit; for instance, in some implementations, absolute priority may not correspond to a single value, but to more than one, for example 0 and 1; this change proves necessary particularly in the case of high operating frequencies, for which freeze instructions issued during cycle t act on the rest of the machine no sooner than cycle t + 2. In addition and independently, it is possible to simplify the implementation when the cycle time Tc of the physical banks is equal to 1; under this assumption, each channel manager, for example GCO_0, is reduced to its sequencer ORO_0, while inputs ADE and PAE, output USS and gates P_1, P_2, P_3, which have become unnecessary, are eliminated.

Claims

1. A system for arbitrating requests and resolving conflicts related to access to the independent-bank memories of computing machines of the type comprising at least one processor and a shared memory including at least one logical bank (BLO) constituted in turn by at least one physical bank (BPO_0, BPO...BPO_{N-1}), linked by independent access channels adapted to operate simultaneously to transfer up to one data item per clock cycle, said channels each comprising a write bus (BUE), a read bus (BUL) and a control bus (BUC) supplying in particular a priority data item issued by the processor, said system being characterized in that it comprises a controller associated with each logical bank to arbitrate conflicts between the requests for access to that logical bank and one freeze instruction cable (CGP) per processor to deliver a freeze instruction signal designed to interrupt the issuing of requests by the processor, in that said controller includes, for each channel, an interface comprising buffers (TCO_0), to wit, write and control input buffers (TEO) and read output buffers (TSO), connected by buses to the channel and to the logical bank between which they are inserted, a write/read manager (GEO_0) connected to the buffers to manage the transfers between them and, on the one hand, the channels, and on the other hand, the logical bank, to the control bus, and to the freeze instruction cable, and a decoder (DEO_0) connected to the control bus and to the write/read manager to indicate to the latter whether requests are intended for the logical bank, and, associated with said interface, which generates an identification tag for each request intended for it, an arbiter (ARO) connected to the write/read managers and to the freeze instruction cable(s) to select from the requests the one that will access the logical bank and, if at least one request having an absolute priority is not satisfied, to order a freeze of the processor concerned, and a delay unit (REO) associated with the logical bank and connected to the write/read managers to propagate control data.

2. The system as in claim 1, characterized in that said arbiter (ARO) comprises, on the one hand, a channel manager (GCO_0, GCO_1...GCO_3) associated with each channel, including a sequencer (ORO_0, ORO_1...ORO_3) connected to the corresponding write/read manager (GEO_0, GEO_1...GEO_3) to receive therefrom the attributes of the incoming requests, and, on the other hand, a selector (ELO) connected to said channel managers to perform the selection of the request intended for memory on the basis of the data from said channel manager to calculate the freeze instructions.

3. The system as in claim 1, characterized in that said arbiter (ARO) comprises, on the one hand, a channel manager (GCO_0, GCO_1...GCO_3) associated with each channel, including a sequencer (ORO_0, ORO_1...ORO_3) connected to the corresponding write/read manager (GEO_0, GEO_1...GEO_3) to receive therefrom the attributes of the incoming requests, an estimator (ESO_0, ESO_1...ESO_3) associated with each sequencer to receive a representation of the status of the physical banks and to detect the risk of conflict, and gates (P_1, P_2, P_3) whose inputs are connected to outputs of the channel manager and of the estimator, and, on the other hand, a selector (ELO) connected to the outputs of said gates, to the channel managers and to the estimators, to perform the selection of the request intended for memory on the basis of the data emanating from these elements of the channel manager, to transmit update data to the estimators, and to calculate the freeze instructions.

4. The system as in claim 2 or 3, characterized in that said arbiter (ARO) comprises inputs (ENE, ETE, PIE, ADE, PAE) connected to outputs of the write/read managers (GEO_0, GEO_1...GEO_3) to receive therefrom, respectively, validation signals, identification tag signals, initial priority signals, signals indicating the number of the physical bank to which the request pertains, and, if estimators are present, signals giving an encoded description of the bytes accessed in the physical bank, as well as outputs (EXS, ETS), also connected to the write/read managers, to supply thereto read validation and read address signals, respectively, and a freeze terminal connected to the freeze instruction cable (CGP).

5. The system as in claim 3, characterized in that said selector (ELO) comprises a circuit for generating a freeze instruction signal (GEL) and comprising gates whose inputs are connected in pairs to corresponding outputs (AGS) of channel managers (GCO_0, GCO_1...GCO_3) to receive therefrom signals to accelerate the calculation of the freeze request, and a circuit for generating a local freeze signal (RG) for the interface blocks, as well as a selection logic comprising, for each channel (C_0, C_1,...C_3), comparators connected in pairs to corresponding outputs (PRS) of the channel managers to receive therefrom signals indicating the current priority, and an array of gates some of which have their inputs connected respectively to the outputs of said comparators and to outputs (CES) of the channel managers to receive therefrom a priority validity signal, with a view toward generating a validation signal (REX) and an extraction signal indicating that the proposed request can be passed to memory and designed to be applied to an extraction input (EXE) of the channel manager.

6. The system as in claim 5, characterized in that said selector (ELO) comprises counters composed of circularly connected registers, each connected to a carry input of a comparator.

7. The system as in claim 1, characterized in that each channel (C_0) is connected to a logical bank (BLO) by means of said write bus and said control bus, cascade-connecting the channel to a demultiplexer (DM_1O), said demultiplexer to the input buffer (TEO), said input buffer to a three-state amplifier (SP_1O), and said three-state amplifier to the logical bank, and by means of the read bus, cascade-connecting said logical bank to the output buffer (TS), said output buffer to an alignment/multiplexing logic (LOO), said alignment/multiplexing logic to another three-state amplifier (SP_2O), and said three-state amplifier to the channel.

8. The system as in claim 7, characterized in that said decoder (DEO_0) has its inputs connected to a demultiplexer (DM_2O) connected to the channel (C_0) and its outputs connected to a parallel-loaded offset register system designed to offset a control word controlling a read amplifier for reads from the output buffer (TSO).

9. A method for arbitrating requests and resolving conflicts related to access to independent-bank memories of computing machines of the type comprising at least one processor and a shared memory including at least one logical bank (BLO) constituted in turn by at least one physical bank (BPO_0, BPO...BPO_{N-1}), linked by independent access channels (C_0, C_1...C_{p-1}) adapted to function simultaneously to transfer up to one data item per clock cycle, said channels each comprising a write bus (BUC) [sic], a read bus and a control bus supplying in particular an initial priority data item issued by the processor and constituted by the number of cycles for which the request can be delayed, said method being characterized in that on each arrival of a request via a channel it is determined for each logical bank whether the request is intended for said logical bank, and if so, the parameters of that request are stored temporarily; on each cycle, from all the requests that have previously been stored and have not yet accessed the physical banks a request is chosen that can be passed on immediately without causing a conflict and is given access to the physical bank, and in the absence of an unsatisfied request having an absolute priority, the priority of all the other still-pending requests is decremented, whereas in the presence of at least one unsatisfied request having an absolute priority, a temporary freeze signal (GEL) is sent via the freeze instruction cable(s) connected to the processor(s) having unsatisfied requests with an absolute priority so as to interrupt the issuing of new requests by said processor(s).

10. The method as in claim 9, characterized in that on the temporary freezing of the processor, access to the physical bank is given to a request that can be passed on immediately without causing a conflict.

11. The method as in either of claims 9 and 10, characterized in that, to arbitrate access to memory, the request having the lowest current priority is selected for each sequencer (ORO_0, ORO_1, ORO_2, ORO_3), said request is tagged with a validity signal (CE), and the valid request having the lowest priority is chosen from among the tagged requests.

12. The method as in either of claims 9 and 10, characterized in that if a request having an absolute priority has not been satisfied, the decrementation of the requests corresponding to all the logical banks (BLO) is interrupted, but the pending requests continue to be streamed normally to the physical banks (BPO_0, BPO_1...BPO_{N-1}).

KEY TO FIGURES:

Fig. 1:

C = channel

D = tap

CO = controller

L = line

BP = physical bank

BL = logical bank

Fig. 2:

BUC = control bus

BUE = write bus

BUL = read bus

C [C_{p-1}, etc.] = channel

TCO = buffer

GEO = read/write manager

DEO = decoder

CGP = freeze instruction cable

BPO = physical bank

Octet = byte

REO = delay unit

ARO = arbiter

GEL = FREEZE

EXS = extraction output

ETS = tag output

ENE = validation input

ETE = identification tag input

PIE = initial priority input

ADE = physical bank number input

PAE = description input

Fig. 3:

ORO = sequencer

ENE = validation input

ETE = identification tag input

PIE = initial priority input

ADE = physical bank number input

PAE = description input

ETS = tag output

EXE = extraction input

AGS = accelerate freeze output

DGS = freeze instruction output

PUE = enable decrementation input

PRS = current priority indication output

VAS = vacancy indication output

USS = field juxtaposition output

GCO = channel manager

P_1, etc. = gate

ESO = estimator

IDE = identification input

COE = parameterization input

MJE = update input

CCS = conflict found output

ECE = mode switching input

MCK = first clock signal

SCK = second clock signal

EXS = extraction output

PRE = priority indication input

CEE = request-validity input

HRE = request-interrupt input

USE = field juxtaposition input

AGE = accelerate freeze input

ELO = selector

GEL = FREEZE

MJS = update output

MCK = first clock signal

SCK = second clock signal

Fig. 4:

PU = decrementation signal

PI = initial priority signal

EN = validation signal

ET = tag signal

AD = physical bank number signal

PA = description signal

EX = extraction signal

Lecture ordonnanceur = sequencer read

Lecture estimateur = estimator read

Ecriture ordonnanceur = sequencer write

Ecriture estimateur = estimator write

AG = accelerate freeze request signal

ET = tag signal

HR = request-interrupt signal

US = field juxtaposition signal

CE = priority validity signal

PR = current priority indication signal

MCK = first clock signal

SCK = second clock signal

Entrées = inputs

Sorties = outputs

Fig. 5:

RG = local freeze signal

SCK = second clock signal

D = flipflop

Q = [presumably] flipflop output

GCO = channel manager

HRS = request-interrupt output

RZ = reset-to-zero signal

RAZ = reset-to-zero terminal

PRS = priority indication output

PUE = decrementation signal input

MJE = update input

AGS = accelerate calculation of freeze request output

EXE = extraction input

USS = field juxtaposition signal output

CES = priority validity output

R_e = input carry

R_s = output carry

REX = validation signal

MCK = first clock signal

GR = freeze instruction signal

GEL = FREEZE

SCK = second clock signal

"RE" (lower right corner): The text seems to indicate that this is an error for RG, local freeze signal. (See bottom of p. 25 of this translation.) – Trans.

Fig. 6:

MCK = first clock signal

RG = local freeze signal

CP_2O, CP_1O = modulo d counters

R_e: input carry

Mem L + 2 = second operand of counter

Priorité = priority

LEE = read/write control input

ALE = read address inputs

AEE = write address input

EEE: Apparently another error for EVE, write validation input. (See Footnote 2, p. 28. – Trans.

TSO = output buffer

Sorties = outputs

Entrées = inputs

LOO = alignment/multiplexing logic

SP_2O = three-state amplifier

DM_2O, DM_1O = demultiplexers

DEO = decoder

VDD: [no explanation found]

ETE: tag input

ENE: validation input

ETS = tag output

ADE = physical bank number input

PAE = description input

GCO = channel manager

PIE = initial priority input

CO = controller

PAO = combinatorial circuit

DCO = decoder

PAS = description output

LOS = physical bank number output

EVE = write validation input

AEE = write address input

ALE = read address input

LEE = read/write control input

SCK= second clock signal

TEO = input buffer

ELO = selector

SP_1O = three-state amplifier

BLO = logical bank

REX = validation signal

[Logo] European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 90 40 0148

DOCUMENTS CONSIDERED MATERIAL

Category	Citation of document, stating relevant portions where appropriate	RE: Claim	CLASSIFICATION OF APPLICATION (Int. Cl. [5])
A	EP-A-0 004 775 (A. H. CROXON) * entire document *	1, 9	G 06 F 13/18
A	IEE PROCEEDINGS SECTIONS A A I Vol. 139, No. 4, Part E, July 1983, pages 116-124, Old Woking, Surrey, GB; E. L. DAGLESS et al.: "Shared memories in the CYBA-M multimicroprocessor" * entire document *	1, 9	
A	EP-A-0 242 882 (HITACHI LTD.) * entire document *	1, 9	
A	PATENT ABSTRACTS OF JAPAN Vol. 6, No. 88 (P-118)(966), 26 May 1982; & JP - A - 57 025054 (NIPPON DENKI K.K.) 09.02.1982 * entire document *	1, 9	

TECHNICAL FIELDS SEARCHED (Int. Cl. 5)

G 06 F 13/18

This report has been drawn up for all the claims.

Place of search	Search completion date	Examiner
Berlin	March 28, 1990	DURAND J.

CATEGORIES OF DOCUMENTS CITED

X: particularly relevant in and of itself
Y: particularly relevant in combination with another document in the same category
A: technological background
O: non-written disclosure
P: interposed document

T: theory or principle on which the invention is based
E: patent document whose date is prior to the filing date and which was not published until said filing date or thereafter
D: cited in the application
L: cited for other reasons
&: member of the same family, corresponding document